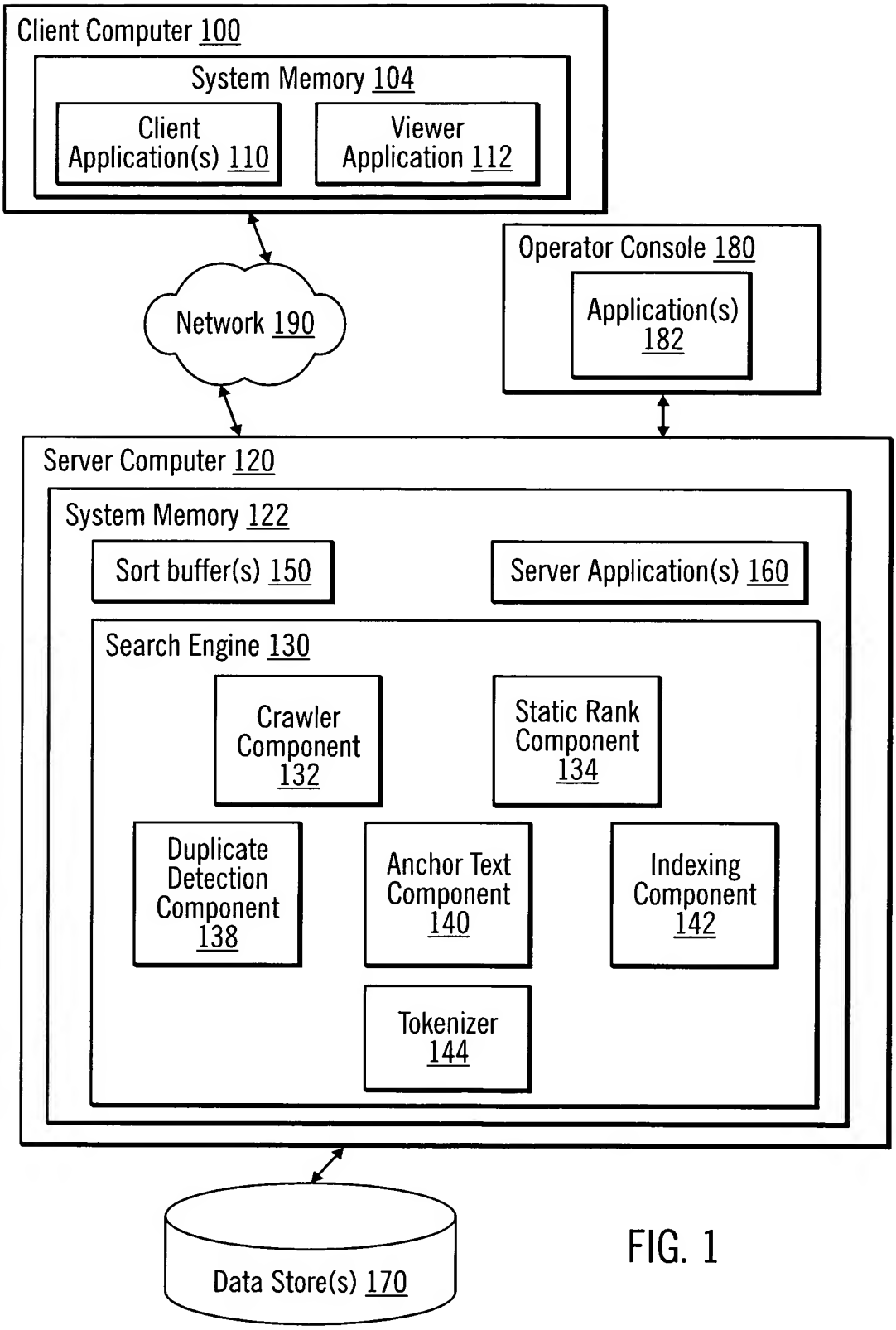
Client Computer 100
System Memory 104
Client Application(s) 110
Viewer Application 112
Network 190
Operator Console 180
Application(s) 182
Server Computer 120
System Memory 122
Sort buffer(s) 150
Server Application(s) 160
Search Engine 130
Crawler Component 132
Static Rank Component 134
Duplicate Detection Component 138
Anchor Text Component 140
Indexing Component 142
Tokenizer 144
Data Store(s) 170

FIG. 1

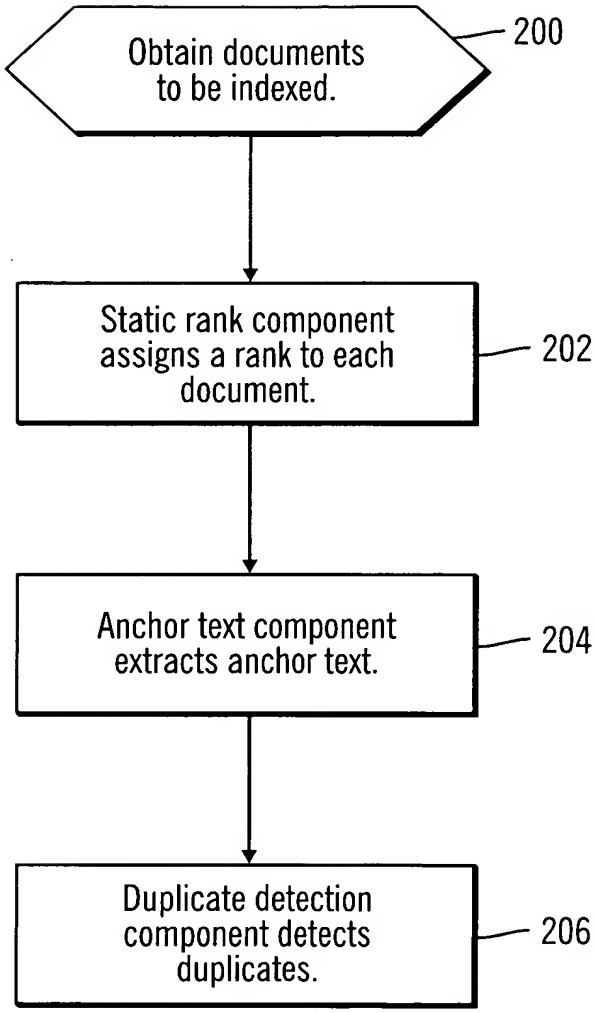
Obtain documents to be indexed.
200
Static rank component assigns a rank to each document.
202
Anchor text component extracts anchor text.
204
Duplicate detection component detects duplicates.
206

FIG. 2

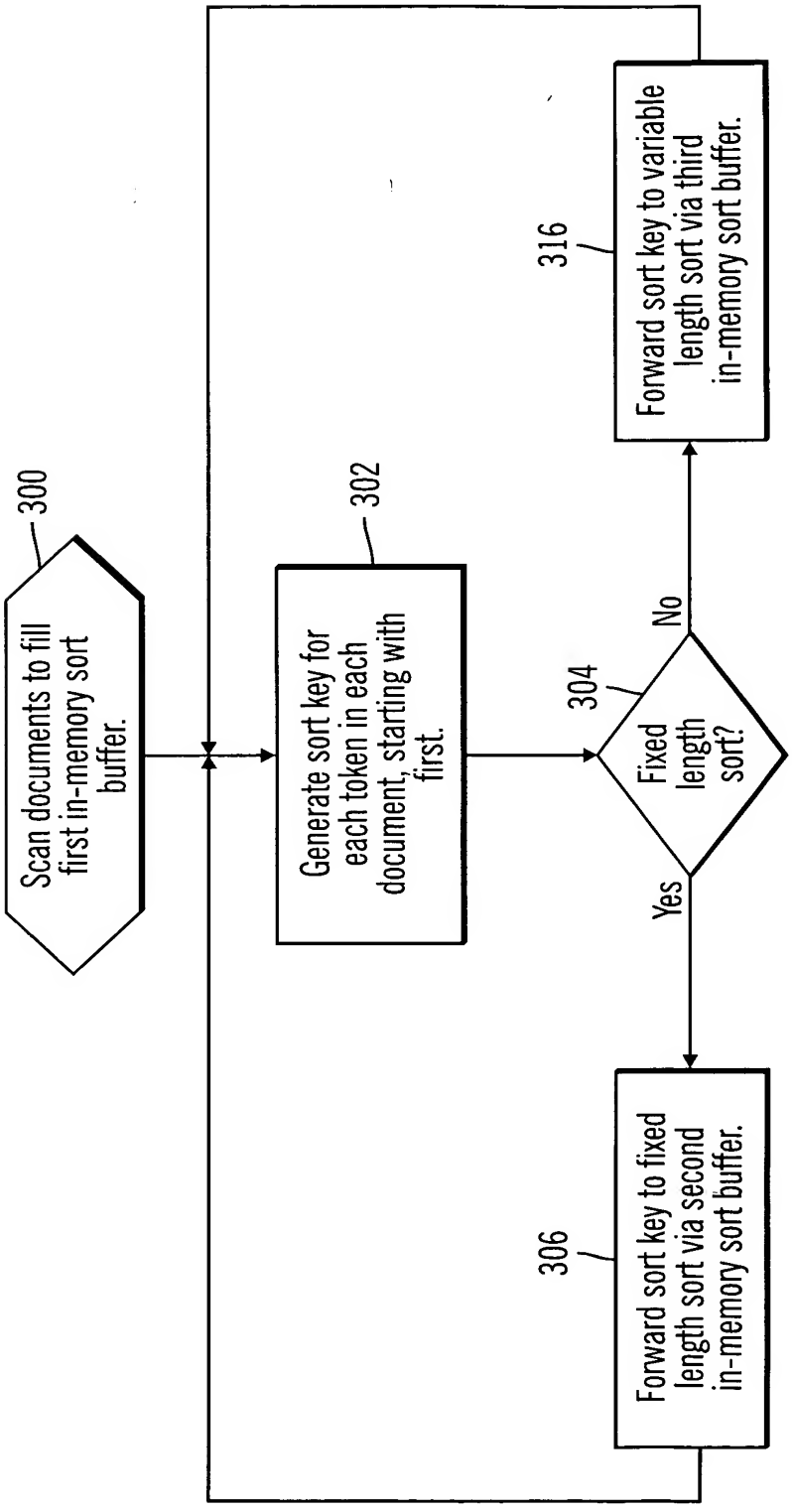
300
Scan documents to fill first in-memory sort buffer.
302
Generate sort key for each token in each document, starting with first.
304
Fixed length sort?
No
Yes
316
Forward sort key to variable length sort via third in-memory sort buffer.
306
Forward sort key to fixed length sort via second in-memory sort buffer.

FIG. 3A

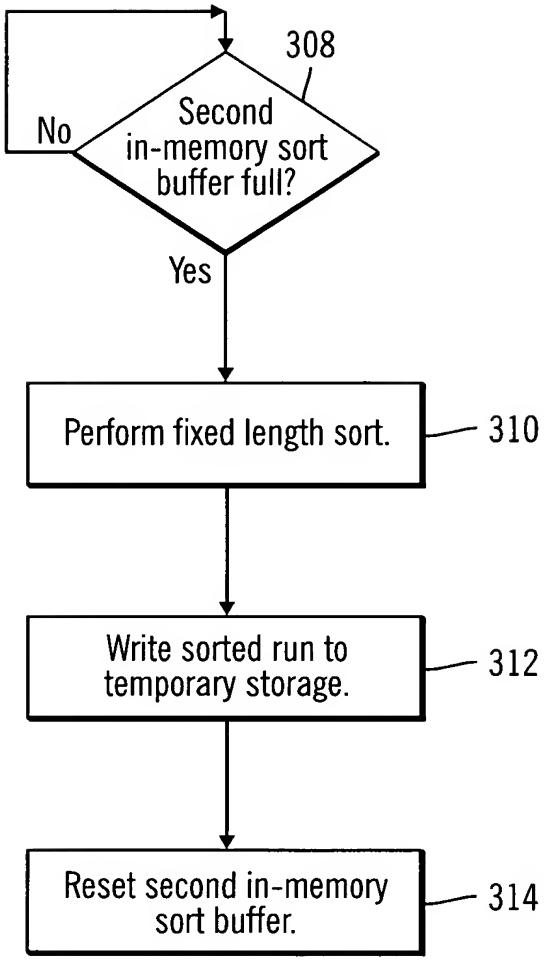
308
Second in-memory sort buffer full?
No
Yes
Perform fixed length sort.
310
Write sorted run to temporary storage.
312
Reset second in-memory sort buffer.
314

FIG. 3B

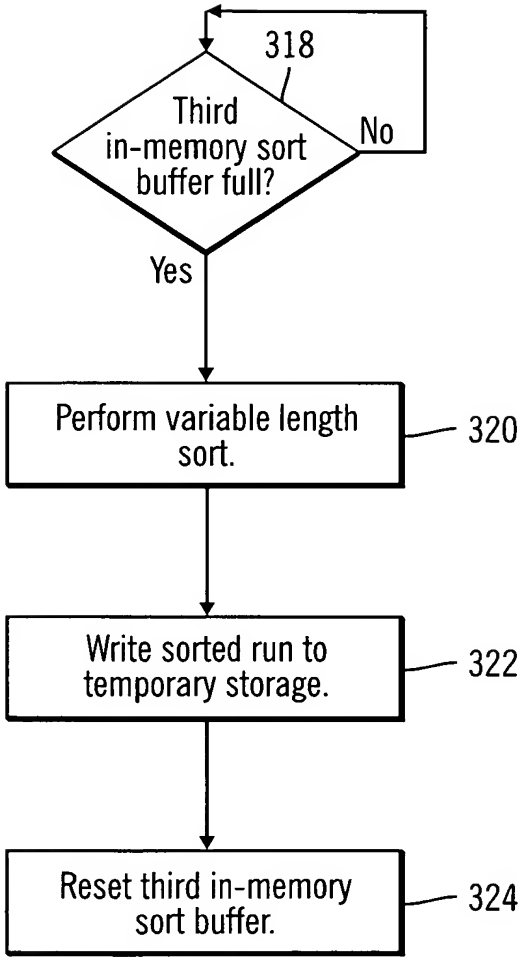
318
Third in-memory sort buffer full?
No
Yes
Perform variable length sort.
320
Write sorted run to temporary storage.
322
Reset third in-memory sort buffer.
324

FIG. 3C

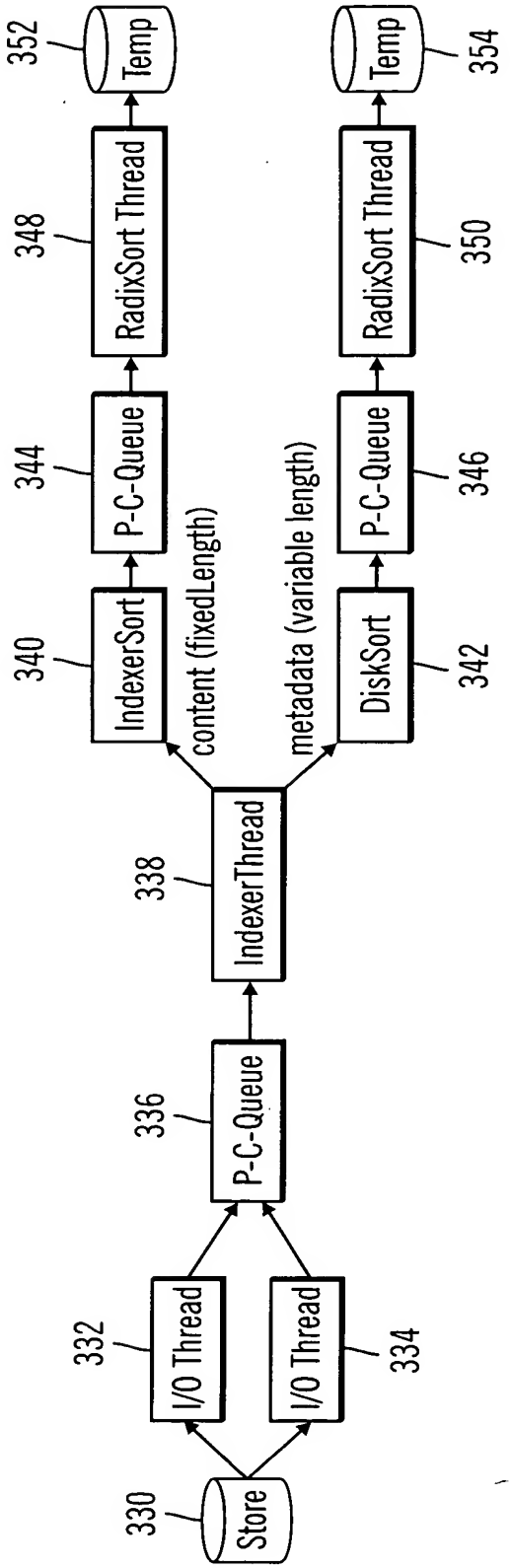
Store
330
I/O Thread
332
I/O Thread
334
P-C-Queue
336
IndexerThread
338
content (fixedLength)
metadata (variable length)
IndexerSort
340
DiskSort
342
P-C-Queue
344
P-C-Queue
346
RadixSort Thread
348
RadixSort Thread
350
Temp
352
Temp
354

FIG. 3D

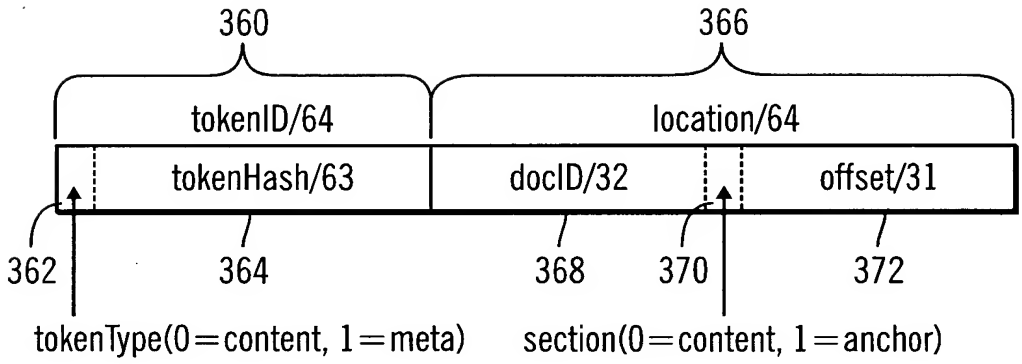
360
366
tokenID/64
location/64
tokenHash/63
docID/32
offset/31
362
364
368
370
372
tokenType(0=content, 1=meta)
section(0=content, 1=anchor)

FIG. 3E

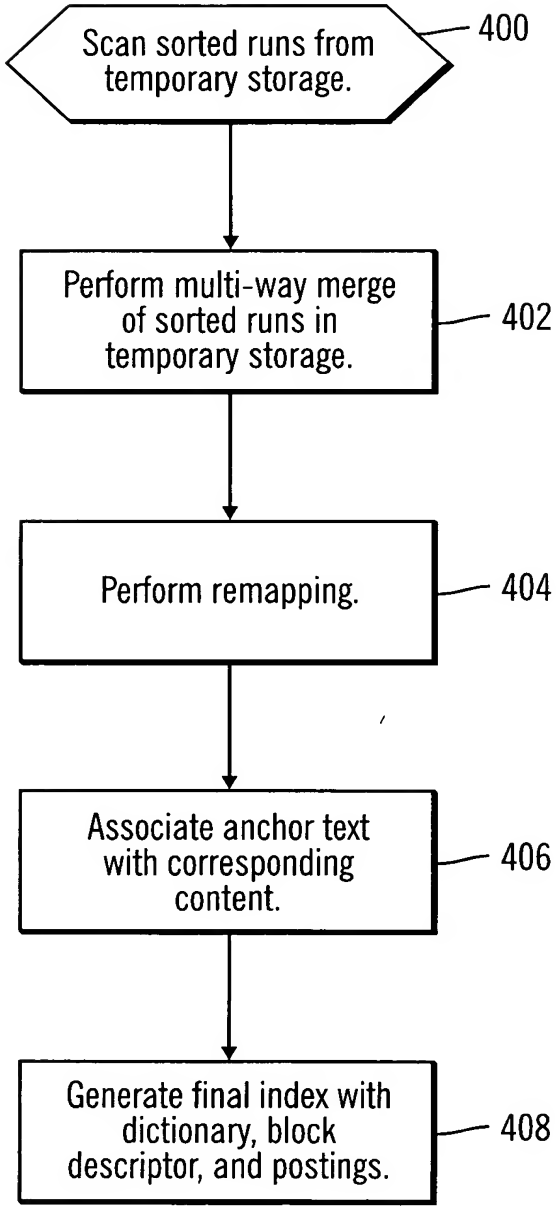
Scan sorted runs from temporary storage.
400
Perform multi-way merge of sorted runs in temporary storage.
402
Perform remapping.
404
Associate anchor text with corresponding content.
406
Generate final index with dictionary, block descriptor, and postings.
408

FIG. 4A

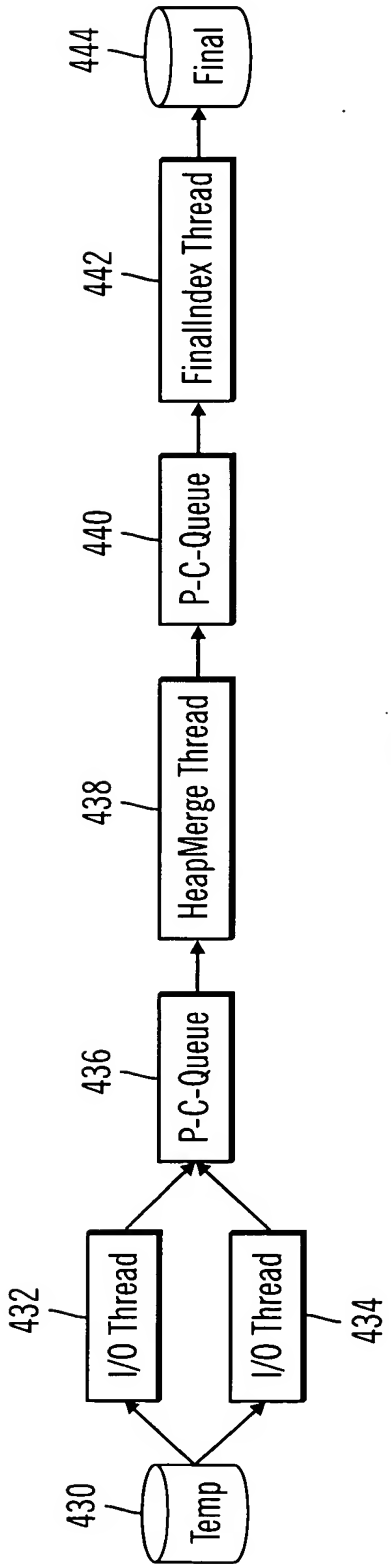
444
Final
442
FinalIndex Thread
440
P-C-Queue
438
HeapMerge Thread
436
P-C-Queue
432
I/O Thread
434
I/O Thread
430
Temp

FIG. 4B

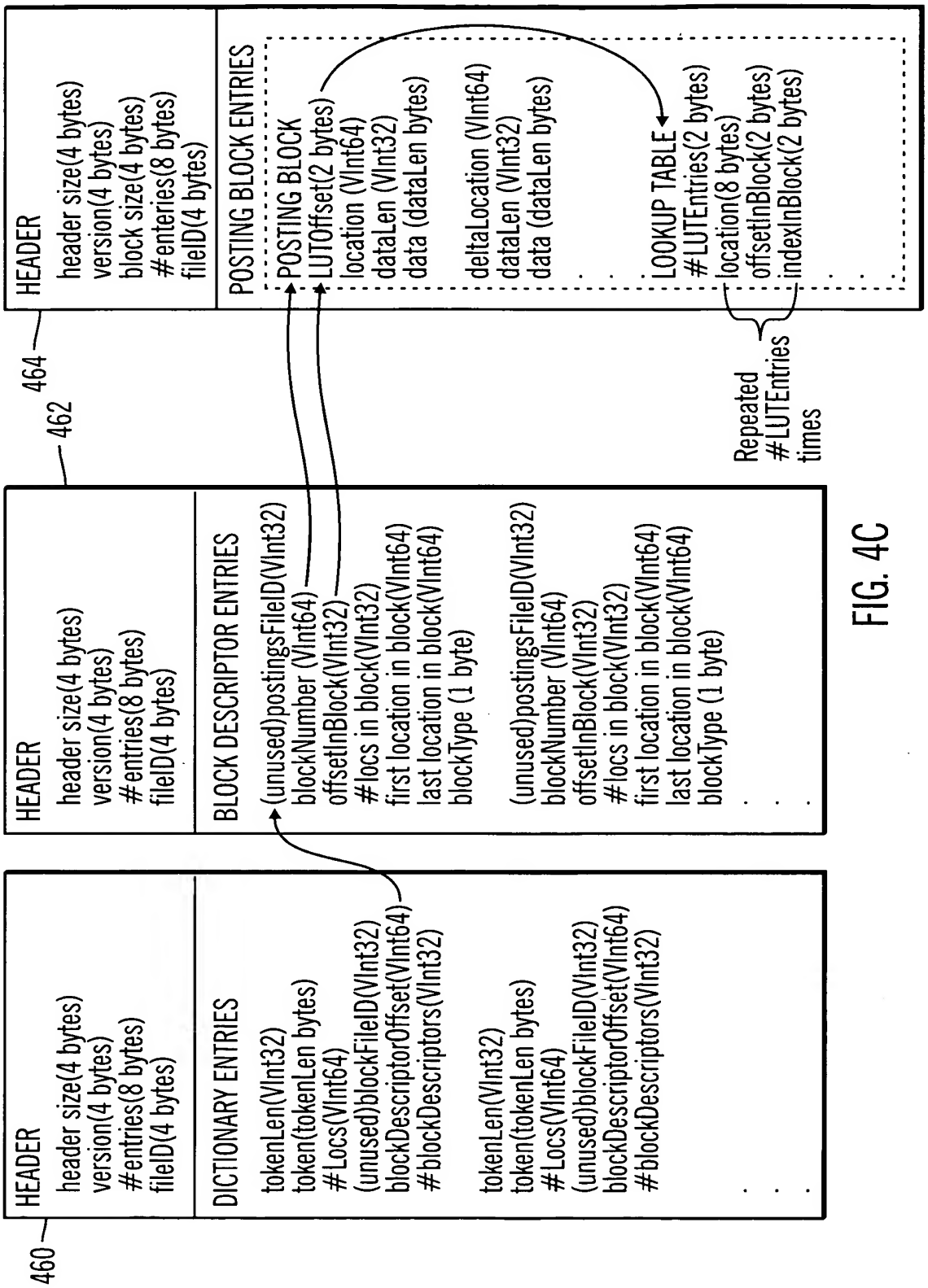
460
HEADER
header size(4 bytes)
version(4 bytes)
#entries(8 bytes)
fileID(4 bytes)
DICTIONARY ENTRIES
tokenLen(VInt32)
token(tokenLen bytes)
#Locs(VInt64)
(unused)blockFileID(VInt32)
blockDescriptorOffset(VInt64)
#blockDescriptors(VInt32)
tokenLen(VInt32)
token(tokenLen bytes)
#Locs(VInt64)
(unused)blockFileID(VInt32)
blockDescriptorOffset(VInt64)
#blockDescriptors(VInt32)
462
HEADER
header size(4 bytes)
version(4 bytes)
#entries(8 bytes)
fileID(4 bytes)
BLOCK DESCRIPTOR ENTRIES
(unused)postingsFileID(VInt32)
blockNumber (VInt64)
offsetInBlock(VInt32)
#locs in block(VInt32)
first location in block(VInt64)
last location in block(VInt64)
blockType (1 byte)
(unused)postingsFileID(VInt32)
blockNumber (VInt64)
offsetInBlock(VInt32)
#locs in block(VInt32)
first location in block(VInt64)
last location in block(VInt64)
blockType (1 byte)
464
HEADER
header size(4 bytes)
version(4 bytes)
block size(4 bytes)
#enteries(8 bytes)
fileID(4 bytes)
POSTING BLOCK ENTRIES
POSTING BLOCK
LUTOffset(2 bytes)
location (VInt64)
dataLen (VInt32)
data (dataLen bytes)
deltaLocation (VInt64)
dataLen (VInt32)
data (dataLen bytes)
LOOKUP TABLE
#LUTEntries(2 bytes)
location(8 bytes)
offsetInBlock(2 bytes)
indexInBlock(2 bytes)
Repeated #LUTEntries times

FIG. 4C

500

CPUs	IndexBuildRun(s)	IndexMerge(s)	Speedup versus baseline
1	617	486	0 %
1+1HT	521	417	18 %
2	421	318	49 %
2+2HT	422	343	44 %

FIG. 5A

510

Documents	Index Size	IndexBuildRuns(s)	IndexMerge(s)
200,000	1.32GB	166.7	154.8
500,000	3.14GB	417.5	341.1
2,000,000	12.3GB	1,713.7	1,345.1
6,000,000	33.4GB	4,632.5	3,775.8

FIG. 5B

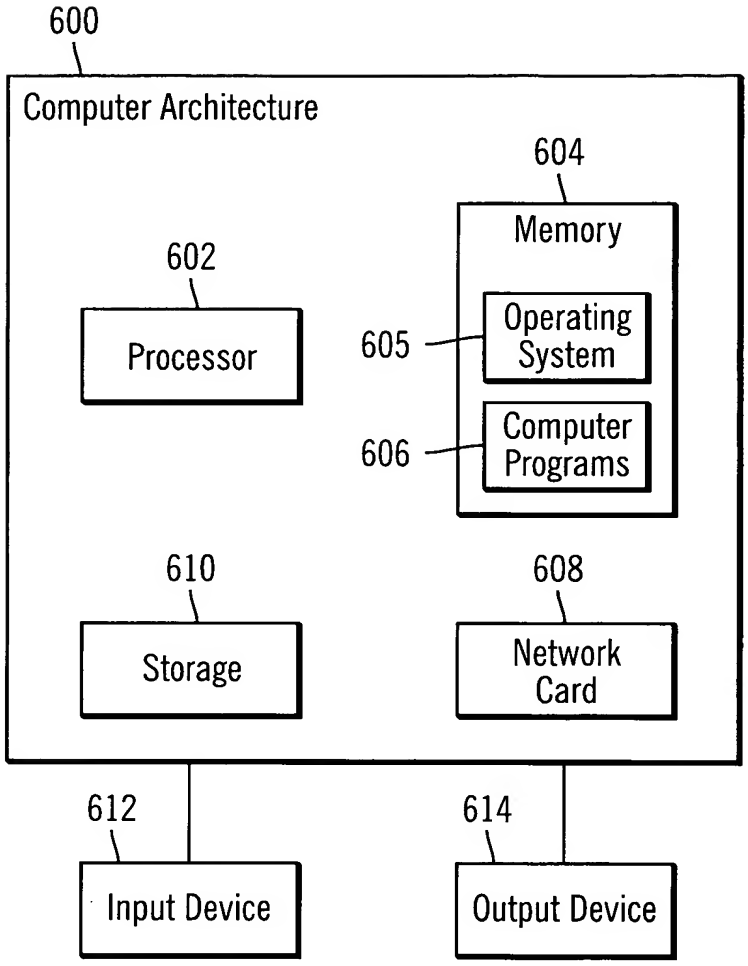
600
Computer Architecture
604
Memory
602
Processor
605
Operating System
606
Computer Programs
610
Storage
608
Network Card
612
Input Device
614
Output Device

FIG. 6